Implementation and Algorithms for Vertex QT-DSM Tree
RHIC 2012 200GeV Heavy Ion Run

Eleanor Judd

May 9, 2012

Change Log:

Date	Description
May 5, 2011	First change for 2011 200 GeV AuAu data taking. The VT201 algorithm has been changed to include preceded protection for the minimum bias bit.
May 25, 2011	The VT201 algorithm has been changed again. The preceded protection has been removed from the primary minimum bias output bit. The secondary minimum bias bit has been replaced with that preceded protection bit. The combinations can now be done at the TCU level. At the same time, a minor bug in the atomcules logic has been fixed.
April 18, 2012	First change for 2012 Heavy Ion running. There is a new ZDC QT algorithm with many more thresholds so there are new ZD101 and VT201 algorithms to deal with them.
May 9, 2012	Changed ZDC logic again for the CuAu running. The ZDC QT algorithm stays the same. Some of the threshold bits are dropped in ZD101 to make room for the "good TAC" bits. The ZDC-related combinations in VT201 are then changed.

The Vertex branch of the DSM tree is used to locate the primary vertex of the RHIC beam collisions at STAR. All three relevant trigger detectors connect to this branch: Zero Degree Calorimeters (ZDC), Beam-Beam Counters (BBC) and the Vertex Position Detector (VPD). The raw detector signals are digitized and pre-processed in QT boards. The DSM tree is then used to calculate TAC differences and combine ADC information to produce (for example) minimum bias or ultra-peripheral triggers.

1. Layer 0 QT Boards: BBQ_BB001:002

There are two BBC small-tile QT boards: one processes data from the East side of the detector and the other from the West side. Please see documentation provided by Chris Perkins for a description of this algorithm

2. Layer 1 DSM Boards: BBC_BB101

The BB101 DSM board processes data from the BBC-small-tile detector. The algorithm receives ADC-sum and fastest-TAC data from the QT boards. The ADC sums are compared to thresholds. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb101_2009_a.rbt

Users: BB101

Inputs: Ch0/1 = QT Board BB001 (East)
Ch2/3 = QT Board BB002 (West)
Ch4/7 = Unused

From each QT board:
bits 0:15 = ADC-Sum
bits 16:27 = Max TAC (Value of zero implies NO good hits)

LUT: 1:1

Registers:
Four registers, all thresholds can be set independently
R0: BBCsmall-EastADCsum_th (16 bits)
R1: BBCsmall-WestADCsum_th (16)
R2: BBCsmall-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
…
5 => select bits 5:11
R3: BBCsmall-WestTAC-select (3)
Same value definitions as for R2

Action:
1st Latch input

2nd Compare each ADC-sum to its threshold
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R2) and TAC-W

3rd Delay ADC-sum threshold bits
Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference
Use R2 to select the TAC-E scaler bits:
If (R2 = 0) then chose TAC-E-scaler-0
Else if (R2 = 1) then chose TAC-E-scaler 1
Etc…
Do the same for West side, using R3 to control the selection.

4th Latch output

Output to VT201:

(0-12) TAC difference
(13) Unused
(14) ADC-sum-E > th0
(15) ADC-sum-W > th0

Scalers:

(0-6) selected bits of TAC-E
(7-13) selected bits of TAC-W
(14) ADC-sum-E > th0
(15) ADC-sum-W > th0

2. Layer 0 QT Boards: BBQ_BB003

There is just one BBC large-tile QT board and it receives data from both the East and West sides of the detector. See documentation from Chris Perkins for a description of this algorithm.

3. Layer 1 DSM Board: BBC_BB102

The BB102 DSM board processes data from the BBC-large-tile detector. The algorithm receives a hit flag and fastest-TAC data for each of the East and West sides of the detector from the QT board. The hit flags indicate there was at least one good hit on each side, and they are just passed through to the output. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb102_2010_b.rbt

Users: BB102

Inputs: Ch0/1 = QT Board BB003 (East and West)
Ch2/7 = Unused

From the QT board:
bits 0:11 = MAX TAC East (value of zero implies no good hits)
bits 12:23 = MAX TAC West
bit 24 = East hit
bit 25 = West hit

LUT: 1:1

Registers:

R0: BBClarge-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
…
5 => select bits 5:11
R1: BBClarge-WestTAC-select (3)
Same value definitions as for R0

Action:

1st Latch input

2nd Delay hit bits to 4th step
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R0) and TAC-W (see register R1)

3rd Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference to the 4th step
Use R0 to select the TAC-E scaler bits:
If (R0 = 0) then chose TAC-E-scaler-0
Else if (R0 = 1) then chose TAC-E-scaler-1
Etc…
Do the same for West side, using R1 to control the selection.

4th Latch output

Output to VT201:

(0-12) TAC difference
(13) Unused
(14) East hit
(15) West hit

Scalers:

(0-6) selected bits of TAC-E
(7-13) selected bits of TAC-W
(14) East hit
(15) West hit

4. Layer 0 QT Boards: BBQ_VP001:002

The two VPD QT boards use the same algorithm as is used by the two small-tile BBC QT boards. Please see documentation from Chris Perkins for a description of this algorithm.

5. Layer 1 DSM Board: BBC_VP101

RBT File: bbc_vp101_2009_a.rbt

Users: VP101

Inputs: Ch0/3 = Unused
Ch4/5 = QT Board VP003 (East)

Ch6/7 = QT Board VP004 (West)

The VP101 DSM board receives VPD data from 2 QT boards. The logic needed to do this analysis is the same as that used by the BB101 algorithm. The VP101 algorithm is therefore identical to the BB101 algorithm in every way, except for the input map. Please see the BBC_BB101 documentation above for details of the logic.

6. Layer 0 QT Board: BBQ_ZD001

For the 2012 heavy ion data taking we have a new ZDC QT algorithm which reduces the number of TAC bits to make room for more threshold bits. Please see documentation provided by Chris Perkins for a description.

7. Layer 1 DSM Board: BBC_ZD101

The ZD101 DSM board processes data from the ZDC detector. The algorithm receives TAC data from the QT boards. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the ZDC. The TAC difference is then compared to a window and a bit is set if it is inside that window. The algorithm also receives the results of comparing sums to thresholds. Most of those threshold bits are passed through to VT201 unmodified. Two thresholds, one from each side, are gated with a deadtime signal. This allows the user to zero out those two thresholds for a certain number of RHIC clock ticks after a ZDC coincidence is detected.

RBT File: bbc_zd101_2010_c.rbt

Users: ZD101

Inputs: Ch0/1 = QT Board ZD001
Ch2:7 = Unused

From the QT board:
bits 0:9 = West-1 TAC
bits 10:19 = East-1 TAC
bit 20 = West attenuated sum > threshold-4 (Unused in this algorithm)
bit 21 = West attenuated sum > threshold-5
bit 22 = East attenuated sum > threshold-4 (Unused in this algorithm)
bit 23 = East attenuated sum > threshold-5
bit 24 = West sum > threshold-0
bit 25 = West sum > threshold-1
bit 26 = West sum > threshold-2
bit 27 = East sum > threshold-0
bit 28 = East sum > threshold-1
bit 29 = East sum > threshold-2
bit 30 = West sum > threshold-3 (Unused in this algorithm)
bit 31 = East sum > threshold-3 (Unused in this algorithm)

LUT: 1:1

Registers:

R0: ZDC-TACdiff-min (11 bits)
R1: ZDC-TACdiff-max (11)
R2: ZDC-EastTAC-select (3)
0 => select bits 0:4
1 => select bits 1:5
…
5 => select bits 5:9
R3: ZDC-WestTAC-select (3)
Same value definitions as for R2
R4: ZDC-deadtime (4 bits)

Action:

1st Latch input

2nd Delay 8 threshold bits (th0, 1, 2 and 5) to the 4th step.
Gate a copy of the threshold-0 bits with the "take_next" bit, i.e.
Gated-West-th0 = West-th0 and take_next
Gated-East-th0 = East-th0 and take next
Calculate: TAC difference = 1024 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(5), (6), (7), (8) or (9)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 31
Else TAC-E-scaler-0 = TAC-E(0:4)
Same logic for all possible bit selections from TAC-E (see description of register R2) and TAC-W (see R3)

3rd Delay the gated threshold bits to the 4th step.
Delay Good-TAC-E and Good-TAC-W to the 4th step.
Zero out the TAC difference if either Good-TAC-E = 0 or Good-TAC-W = 0.
Otherwise, compare the TAC difference to the window defined by R0 and R1
R0 < TAC difference < R1
Use R2 to select the TAC-E scaler bits:
If (R1 = 0) then chose TAC-E-scaler-0
Else if (R1 = 1) then chose TAC-E-scaler-1
Etc…
Do the same for the West side using R3 to control the selection.
Check for a ZDC coincidence:
Coincidence = Gated-West-th0 and Gated-East-th0 and
Good-TAC-W and Good-TAC-E
If there is a coincidence then initialize a counter to R4. Allow it to count down to zero at a rate of one count per tick of the RHIC clock. Set the "take_next" bit to zero while the counter is non-zero.

4th Latch output

Output to VT201:

(0) TAC difference in window
(1) Gated-W > th0

(2) Gated-E > th0
(3) ZDC deadtime coincidence
(4) W > th0
(5) W > th1
(6) W > th2
(7) Good-TAC-W
(8) Unused
(9) W-attenuated > th5
(10) E > th0
(11) E > th1
(12) E > th2
(13) Good-TAC-E
(14) Unused
(15) E-attenuated > th5

Scalers:

(0-4) selected bits of TAC-E
(5-9) selected bits of TAC-W
(10) Gated-W > th0
(11) W > th0
(12) Gated-E > th0
(13) E > th0
(14) ZDC coincidence
(15) Unused

8. Layer 2 Vertex DSM Board: L1-VT201

All threshold bits of the Vertex tree from the large and small-tile BBC, the ZDC and the VPD are brought into the Vertex DSM. They are passed on to the TCU, some as individual bits and some in combinations. In parallel the TAC difference values are brought into the Vertex DSM. Windows are placed around each TAC difference, and the "inside window" bits get passed through to the TCU and the scaler system. Some of the TAC difference bits from the BBC small-tiles and the VPD are also in the scaler output. A minimum bias bit, based on an OR of information from all 4 detectors is created. This bit is used to start a counter whose status can be used to provide preceded protection for subsequent triggers.

RBT File: l1_vt201_2012_b.rbt

Users: VT201

Inputs: Ch 0 = BB101
Ch 1 = BB102
Ch 2 = ZD101
Ch 3 = Unused
Ch 4 = VP101
Ch 5:7 = Unused

From Small tile BBC-DSM BB101
(0-12) Small tile TAC-Difference
(13) Unused
(14/15) Small tile ADC East/West sum > th0

From Large tile BBC-DSM BB102
(0-12) Large tile TAC-Difference
(13) Unused
(14/15) East/West hit

From ZDC DSM ZD101
(0) TAC difference in window
(1) Gated-W > th0
(2) Gated-E > th0
(3) Unused
(4) W > th0
(5) W > th1
(6) W > th2
(7) Good-TAC-W
(8) Unused
(9) W-attenuated > th5
(10) E > th0
(11) E > th1
(12) E > th2
(13) Good-TAC-E
(14) Unused
(15) E-attenuated > th5

From VPD-DSM VP101
(0-12) VPD TAC-Difference
(13) Unused
(14/15) VPD ADC East/West> th0

LUT: Either 1-to-1 or TAC-difference range conversion

Registers:
R0: BBCsmall-TACdiff-Min (13 bits)
R1: BBCsmall-TACdiff-Max (13)
R2: BBClarge-TACdiff-Min (13)
R3: BBClarge-TACdiff-Max (13)
R4: VPD-TACdiff-Min (13)
R5: VPD-TACdiff-Max (13)
R6: Minimum-Bias-Select (4)
R7: Min_Bias_Protection_Time (9)

Action

1st Latch inputs

2nd Delay the threshold bits from all four detectors, and three ZDC TAC-related bits to the 3rd step.
Delay a second copy some bits to the 4th step.
Delay a copy of the BBC-small and VPD TAC difference to the 4th step.
Combine the ZDC (un-gated) th0 and th1 bits to make windows on the East and West sides separately, i.e.:
ZDC-E-Window = E>th0 and not E>th1

ZDC-W-Window = W>th0 and not W>th1

Compare each of the 3 TAC differences to its minimum and maximum value, as specified in the relevant registers. The logic looks for the TAC difference to be greater than the minimum and less than the maximum.

3rd Make the following combinations of ZDC and VPD bits:

ZDC-COINC = Gated-W>th0 and Gated-E>th0
ZDC-UPC = ZDC-E-Window and ZDC-W-Window
ZDC-hit-East = E>th0 and Good-TAC-E
ZDC-hit-West = W>th0 and Good-TAC-W
ZDC-let-OR = E>th5 or W>th5

VPD-COINC = VPD-E>th and VPD-W>th

Combine the results of the TAC difference comparisons to determine if each TAC difference is inside its specified window, e.g.:

VPD-Tdiff = R4 < VPD TAC difference < R5

Combine the results of the TAC difference comparisons and the ADC threshold bits to make the minimum bias bit, using R6 to turn each component on/off, i.e.:

MB = (R6(0) and BBC-S-Tdiff and BBC-S-E>th0 and BBC-S-W>th0) or
(R6(1) and BBC-L-Tdiff and BBC-L-E>th0 and BBC-L-W>th0) or
(R6(2) and ZDC-Tdiff) or
(R6(3) and VPD-Tdiff)

The preceded logic is only enabled if R7 is set to a non-zero value. In this case, whenever the minimum bias bit is set a counter is initialized to R7-1. The counter then counts down to zero at a rate of one count per tick of the RHIC clock. If another minimum bias interaction occurs while the counter is counting, then the counter is re-initialized to R7-1 and counting continues. The preceded bit is true whenever the current counter value is non-zero, and false otherwise.

4th Latch Outputs

Output to TCU:

Bit	Name	Description
Bit 0	BBC-TAC	BBC small-tile TAC difference in window
Bit 1	BBC-E	BBC small-tile East ADC sum > threshold
Bit 2	BBC-W	BBC small-tile West ADC sum > threshold
Bit 3	BBC-L-E	BBC large-tile East hit
Bit 4	BBC-L-W	BBC large-tile West hit
Bit 5	ZDC-TAC	ZDC TAC difference in window
Bit 6	ZDC-COINC	Gated ZDC East > th0 AND Gated ZDC West > th0
Bit 7	ZDC-UPC	ZDC East in window (th0 and th1) AND ZDC West in window (th0 and th1)
Bit 8	ZDC-E-th2	ZDC East > th2
Bit 9	ZDC-W-th2	ZDC West > th2
Bit 10	ZDC-hit-E	ZDC East > th0 AND Good ZDC-East-TAC
Bit 11	ZDC-hit-W	ZDC West > th0 AND Good ZDC-West-TAC
Bit 12	ZDC-let-OR	ZDC East > th5 or ZDC West > th5
Bit 13	VPD-TAC	VPD TAC difference in window
Bit 14	VPD-COINC	VPD East ADC sum > threshold AND VPD West ADC sum > threshold
Bit 15	Preceded	Counter started by Minimum Bias bit still counting.

Output to Scalers

Bit	Description
Bit 0	BBC small-tile TAC difference in window
Bits 1:4	4 MSB of BBC small-tile TAC difference
Bit 5	Unused
Bit 6	ZDC TAC difference in window
Bit 7	Gated ZDC East > th0 AND Gated ZDC West > th0
Bits 8	Minimum Bias
Bits 9	Unused
Bits 10	Unused
Bit 11	VPD TAC difference in window
Bits 12:14	3 MSB of VPD TAC difference
Bit 15	Preceeded